Scribble Tracker: A Matting-based Approach for Robust Tracking

Jialue Fan, *Student Member, IEEE,* Xiaohui Shen, *Student Member, IEEE,* and Ying Wu, *Senior Member, IEEE*

Abstract—Model updating is a critical problem in tracking. Inaccurate extraction of the foreground and background information in model adaptation would cause the model to drift and degrade the tracking performance. The most direct but yet difficult solution to the drift problem is to obtain accurate boundaries of the target. We approach such a solution by proposing a novel model adaptation framework based on the combination of matting and tracking. In our framework, coarse tracking results automatically provide sufficient and accurate scribbles for matting, which makes matting applicable in a tracking system. Meanwhile, accurate boundaries of the target can be obtained from matting results even when the target has large deformation. An effective model combining short-term features and long-term appearances is further constructed and successfully updated based on such accurate boundaries. The model can successfully handle occlusion by explicit inference. Extensive experiments show that our adaptation scheme largely avoids model drift and significantly outperforms other discriminative tracking models.

Index Terms—Visual tracking, matting, model updating, video analysis.

1 Introduction

Object tracking is a fundamental task in computer vision. Although numerous approaches have been proposed, robust tracking remains challenging due to the complexity in the object motion and the surrounding environment.

In a practical visual tracking system, once the target location is estimated in the current frame, new foreground/background information would be extracted to update the target model. One critical issue is the degradation of the model caused by the inaccuracy in the estimation of the foreground and background, which is rarely discussed in existing methods. Most commonly, the foreground and background are divided by a bounding box or a region around the location of the target. No matter how tight the region is, such a partition is too rough, because some background regions are treated as part of the foreground, especially when the location of the target is not precise or the target is occluded. Accordingly, the updated model would gradually be degraded, and thus cause drift. When the target boundary is inaccurately presented by a rough bounding box, foreground/background labeling errors are inevitably introduced. Any efforts to alleviate drift conducted in the succeeding tracking modules [15], [2] cannot essentially handle such errors. On the contrary, if an accurate boundary of the target can be obtained, such errors would be mostly reduced. Therefore, an accurate boundary that clearly divides the target from the background is very desirable.

There are several methods to obtain a boundary of the foreground (*e.g.*, active contour, image segmentation). Among these methods, one effective way is to perform matting based on some prior information, which has been shown very successful in estimating the opacity of the foreground. The boundary can then be easily extracted from the opacity map.[1] However, matting has never been combined with tracking before because of the gap that matting needs user interaction while tracking requires automatic processing. In this paper, we bridge this gap by automatically providing suitable scribbles for matting during the tracking process, and thus make matting work very well in the tracking scenario. To reliably track a target in a cluttered background, we employ a simple but practical model for tracking, which captures the information from target and its neighboring context, and includes both short-term and long-term descriptions. Furthermore, we propose a practical model adaptation scheme based on the accurate object boundary estimated by matting. Such an interplay of matting and tracking therefore forms a closed-loop adaptation in an object tracking system, as shown in Fig. 1. As will be shown in the following, scribbles play a key role in tracking. Thus we call our method *scribble tracking*.

Benefiting from the matting results, our model adaptation largely excludes the ambiguity of foreground and background, thus significantly alleviating the drift problem in tracking. Besides, object scaling

• *Jialue Fan, Xiaohui Shen, and Ying Wu are all with Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, 60208.*
Email: jialue.fan@northwestern.edu,
{xsh835, yingwu}@eecs.northwestern.edu.

1. We will discuss the relevance of introducing matting to tracking in Sec. 2.1, after a brief introduction of matting.

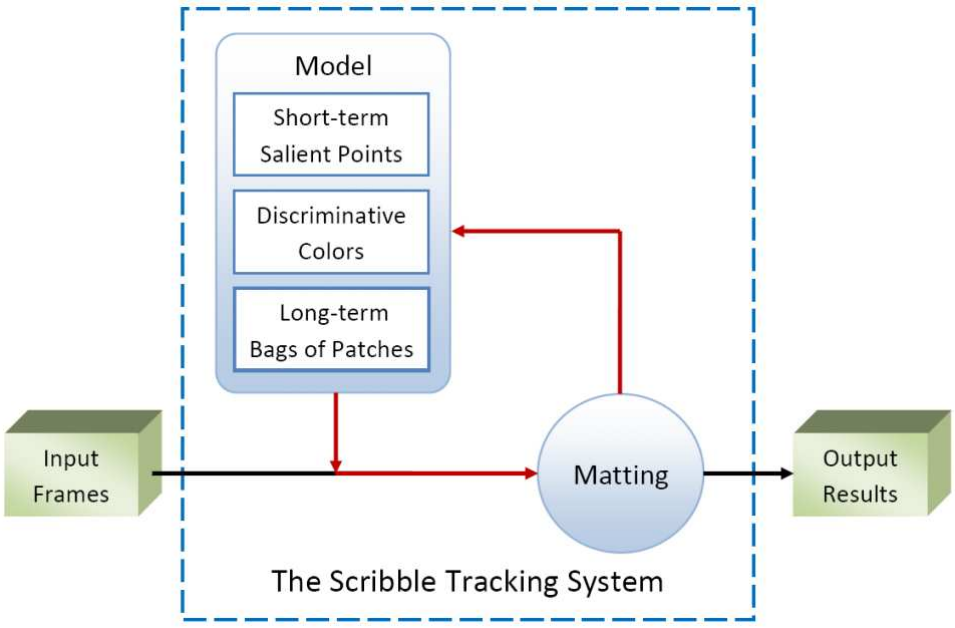


Fig. 1. The framework of scribble tracking.

and rotation can also be handled by obtaining the boundary. As a result, our method can accurately track the target in a long range under various situations, such as fast motion, large deformation, out of plane rotation, and severe occlusion, even when the target reappears after complete occlusion.

Compared to other tracking approaches, our scribble tracking system has the following contributions and advantages:

1) We address the automatic scribble generation problem for matting in the tracking process. A coarse but correct partition of foreground and background is estimated during tracking, which is then used to automatically generate suitable scribbles for matting. The supply of scribbles is non-trivial. A small false scribble may lead to matting failure, while deficient scribbles could also impair the performance. In our system, the generation of scribbles is designed carefully to be correct and sufficient, which can yield comparable matting results to the methods based on user input.
2) Our scribble trimap provides a strong hint on the rough target position, so it can not only be adopted in our method, but can also be considered as a complementary tool for other tracking methods: Since a large number of pixels can be determined as foreground/background using scribbles, one can largely reduce the search space for tracking based on the scribble trimap.

2 Related Work

In this section, we first give a brief introduction of matting, and reveal its relevance to tracking. Then we visit other related tracking methods for comparisons.

2.1 Matting

Image matting is a problem of estimating the opacity of the foreground [9], [27], [41], [3], [40]. Matting methods assume an input image to be a linear composite of a foreground image F and a background image B:

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i, \quad (1)$$

where I_i is the color of the i-th pixel, and α_i is the opacity of i-th pixel's foreground. Since F_i, B_i, and α_i are all unknown, it is a severely under-determined system. Therefore, most matting methods need user input to get the initial α values at some pixels, and then estimate the matte of the entire image based on various assumptions, such as color line model [27] and local pixel correlations [41]. Early methods require users to provide a trimap, which is a rough segmentation of the image into three parts: foreground, background, and unknown regions [9]. More recently, only a small amount of scribbles indicating foreground in white and background in black are required [27], which largely reduces users' interaction. Figure 2 gives one example to illustrate how matting works.

The relevance of introducing matting to tracking is straightforward in the following sense: The root of visual tracking is to match points or regions between the current observation and the target/background model. In the current frame, one point/region can be determined within the target/background if it finds a good correspondence in the target/background model, either by point matching or color matching.

(a) (b) (c)

Fig. 2. An example of matting. (a) The original image. (b) The input image for matting, which is the original image with user specified scribbles (white scribbles indicate the foreground, while black ones indicate the background). (c) The output result (the alpha matte) is the opacity map with values ranging from 0 to 1. Images are taken from [27].

The target location is estimated based on all these correspondences. The matting procedure works in a similar way: Given the user specified scribbles which pre-determine some pixels as the target/background, the algorithm automatically obtains the opacity map (Fig. 2), which clearly separates the target from the background. By filling some gaps such as how to construct the target/background model, how to generate scribbles automatically from the model, and how to perform model updating, etc., one can fit matting naturally into the tracking framework.

The existing works on video matting [8], [4] can not be directly used in tracking because of the requirement of user interaction, which is not suitable in automated tracking methods. Moreover, they can not well handle objects with fast deformations and occlusions, which are very common in most tracking scenarios. To the best of our knowledge, our method is a first attempt to combine matting with tracking to provide shape boundary of the target and to handle occlusion.

2.2 Related tracking methods

Object tracking has been an active research area since early 1980s, and a large number of methods were proposed during the last three decades. A comprehensive overview of these methods can be found in [45]. In the perspective of model design and update in tracking, early works tended to construct the model by describing the target itself [11], [7], [36], [23], while recently the adoption of context information has become very popular [43], [10], [32], [1], [14], [28], [44], [48], [34], [47]. Although these methods more or less suffer from the inaccuracy in the estimation of the foreground and background, they motivate us to design a hybrid appearance model which captures the information from target and its neighboring context, and includes both short-term and long-term descriptions [20], [33], [38], [21].

The drift problem was discussed in [31], in which they proposed a partial solution for template update. In [42], a local generative model was combined in discriminative feature selection to alleviate drift. Grabner *et al.* [15] proposed an online semi-supervised boosting method, and Babenko *et al.* [2] introduced multiple instance learning to avoid the drift of positive samples. These methods all focused on the alleviation of the aftereffects caused by foreground/background labeling errors, but cannot essentially handle these errors. In contrast, such errors would be mostly reduced if an accurate boundary of the target can be obtained.

There are several tracking methods trying to obtain a boundary of the foreground [5], [37], [6], [16], [30], [39]. Tracking using active contours [22] is one way to extract and track the object boundaries [19]. However, active contour tracking heavily relies on curve matching and is not designed for complex shape deformation, therefore cannot handle large deformation. Image segmentation is another direct and popular solution to separate the foreground and background [46]. In [37], a shape-based method was proposed to match the contour of a prior shape to the current image, but contour matching alone may not achieve good results since the useful information within the target boundary is not taken into account. Ren *et al.* [35] combined spatial and temporal cues in a conditional random field (CRF) to segment the figure from the background in each frame. However, some tracking scenes may have cluttered backgrounds, which cause difficulties to directly extract accurate boundaries using segmentation techniques. Besides, CRF models contain many parameters, which are also sensitive to different scenes.

Compared with image segmentation, matting tries to exploit the linear compositing equations in the alpha channel instead of directly handling the complexity in natural images, therefore may achieve better foreground/background separation performance based on a moderate amount of user input. Another advantage which attracts us to use matting is the concept of scribbles, *i.e.*, a set of pixels which are predetermined as the target/background. In our method, the adaptive appearance model automatically generates scribbles in each frame. As the performance of foreground/background separation only relies on the scribbles, it is easy to tune parameters by checking the scribble trimap. On the contrary, in existing image segmentation methods, we do not have such meaningful intermediate results for reference. For example, graph cuts methods do not provide any uncertainty measure associated with the solution they produce [24]. Therefore, tuning parameters becomes relatively hard: Careful analysis is often required to get good segmentations in level-set methods [12] and graph cuts methods. Extensive experiments showed that our boundaries derived from matting are more accurate and robust than the foreground/background separation results generated by other segmentation-based methods such as [35].

3 Model description

For clarity of exposition, we begin by describing our tracking model. By incorporating the properties of discriminative models and descriptive models, our tracking model tries to discriminate the foreground from the background as well as maintain a long-term description for the target's appearance. Apart from the basic dynamics, the model is composed of three main components.

Short-term salient points. S_f denotes a set of salient points that are extracted from the foreground, while S_b is a set of salient points detected from the surrounding background near the target, as shown in Fig. 3(a). Currently, SIFT [29] are used as salient points. Salient points are tracked in a short time period and used to generate scribbles for matting.

Discriminative colors. Color is another useful clue to discriminate the foreground from the background. We select the most discriminative colors for the foreground and the background, respectively. Given a frame with known foreground and background (either by manual initialization at the first frame or by refined tracking results at the following frames), we first get the color histogram of the foreground H_F and the histogram of the background H_B in HSV color space[2]. The log-likelihood ratio of these two histograms can then be easily calculated with respect to each bin:

$$L_i = log(\frac{H_F(i)+\epsilon}{H_B(i)+\epsilon}), \tag{2}$$

where L_i is the log-likelihood ratio of the i-th color bin, ϵ is a very small constant to avoid infinity when H_F or H_B approaches zero. Clearly, when a color mainly appears in the foreground and rarely appears in the background, its L value would be high, and vice versa. Accordingly, we consider the i-th color bin as a distinctive color for the foreground if L_i is larger than a threshold T_L, and consider the color as a distinctive color for the background if $L_i < -T_L$. A discriminative color list of the foreground C_f as well as a similar list of the background C_b are then maintained respectively, which contain the most distinctive colors against each other. Figure 3(b) gives us an example. In Fig. 3(b), the blue color is the most distinctive one for the foreground, while red and gray are distinctive for the background. Black exists in both the foreground and background. Therefore its log-likelihood ratio is near zero, and neither of C_f and C_b chooses it as a discriminative color. Such a description, although simple, is observed very powerful to detect the foreground and the background.

Long-term bags of patches. We also constructed a long-term model to preserve the appearance variation of the target in a long range, which helps locate the target under occlusion. Given a target region (the foreground), we divide it to a $M \times N$ grid. For example, in Fig. 3(c), the grid is 3×3. At each crosspoint of the grid, a **patch** with size $K \times K$ is cropped and recorded. In practice we choose $K = 25$. Therefore, we have many patches with different time stamps at each crosspoint of the grid, which captures the variation in the local appearance at a relatively fixed position of the foreground. We call the set of all the patches at the same crosspoint a **bag-of-patches** BoP_i. For example, in Fig. 3(c), BoP_1 captures the long-term appearance at the top-left corner of the target. The adaptations of those bags are performed independently by foreground matching and occlusion inference, which avoids false update due to partial occlusion. A bag-of-patches not only contains previously appeared patches, but also records their frequency, *i.e.*, their recurrence time. The geometric information (normalized by target size) of these bags of patches is also implicitly encoded by their relative positions.

At each frame, the short-term features in the model are used to generate foreground/background scribbles for matting, and the long-term model is utilized to locate the object when it is under severe occlusion and the short-term features are not reliable. Once the accurate foreground boundary is determined by matting, all the components of the model will be updated accordingly, which will be introduced in the next two sections.

4 From coarse tracking to matting

Given an input frame, we first use our model to perform coarse tracking, *i.e.*, locate the target and obtain a coarse but correct partition of the foreground and background. Based on such a partition, scribbles are automatically generated for matting. The matting results heavily rely on the prior information of the foreground and background. A false labeling of foreground or background may cause a drastic erroneous matte. We carefully design the scribble generation scheme to avoid false labeling and to yield good alpha mattes.

4.1 Short-term coarse tracking

From frame to frame, we use two types of features to detect the foreground and background, and then generate scribbles: salient points and discriminative color regions.

Salient points. Consider that S_f and S_b are the sets of salient points extracted from the foreground and its neighboring background at the previous frame f^{t-1} respectively, and S'_f and S'_b are the corresponding salient point sets at the current frame f^t. First we perform SIFT matching between S_f and S'_f. However, we cannot guarantee that the salient points at f^{t-1} are still salient at f^t. Therefore, for those points in

2. We divide the color to 1024 bins in HSV space (16 bins, 8 bins and 8 bins in the H, S and V channels respectively), and then get the color histogram of a region.

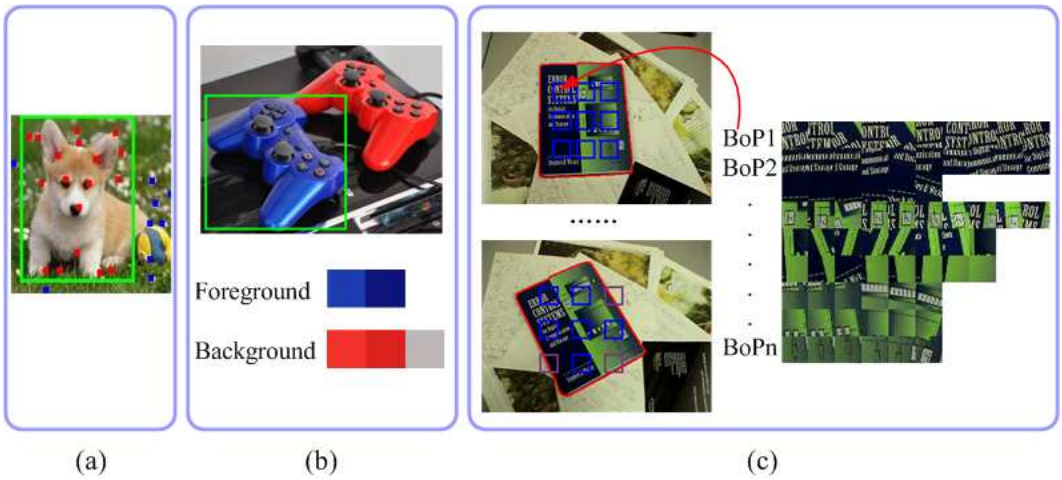


Fig. 3. Our model for tracking. (a) Short-term salient points, (b) discriminative color lists, (c) long-term bags of patches.

S_f that do not find their matching points in S'_f, they are tracked by calculating SSD and gradient-based search to find new locations at f^t [17]. At last, all the points in S_f will have matched locations at the current frame. Small image regions that cover these new locations are then labeled as the foreground. Similarly, we track all the salient points in S_b and label the regions covering their new locations as the background, as we can see in Fig. 4(b). The sizes of scribbles depend on the positions of salient points and their matching scores. If a salient point is far from the object boundary at f^{t-1} and its matching score is relatively high, the corresponding scribble will be relatively large, otherwise it will be small to avoid false labeling.

It is worth notice that in our coarse tracking process, the tracking results of these salient points are not necessary to be very accurate. *It only requires that S_f still stay in the object and S_b remain in the background*, which is robust to some fluctuations in salient point tracking. In our experiments, we found that such requirements are easily satisfied by the tracking results.

Discriminative color regions. Although the regions with salient points are labeled, there are still large uncertain regions around the object, some of which are very discriminative in color space. Therefore, we choose discriminative color regions as additional scribbles. Consider that the discriminative color lists C_f and C_b have been updated at f^{t-1}. At f^t, we use these two lists to detect foreground discriminative color regions and background regions at the possible locations of the target. For each pixel, if its color is the same as one color in foreground discriminative color list C_f, it will be labeled as foreground. Similarly, the pixels with the same colors as in C_b are marked as background, as shown in Fig. 4(c).

The scribbles provided by salient points and the ones provided by discriminative color regions are good complements to each other (see Fig. 3 [13] for one example). Combing two categories of scribbles, the final scribbles for matting are drawn in Fig. 4(d), which ensures to produce a satisfying matte.

Given such scribbles, standard matting methods can be adopted to estimate the matte of current frame. Here we use the closed-form solution proposed in [27]. As we can see in Fig. 4(d), Our scribbles are already good and sufficient to estimate a good matte, which is very competitive against the matting result based on user input in Fig. 4(e).

4.2 Long-term target locating

In most situations, the tracking results of salient points and the identification of discriminative colors are satisfying to help generate a good alpha matte. However, in some cases, such a method is not applicable, especially when the target is severely occluded and no sufficient salient points can be provided, or when the target reappears from complete occlusion. To address those problems, we use our long-term bag-of-patches model to locate the target.

Our model matching approach is based on an underlying assumption: no matter whether the target is severely occluded or first reappears, only a small part of the target is visible. Therefore, only one or two bags-of-patches in our long-term model are in the foreground at this time. That assumption significantly simplifies our matching scheme. We can independently detect each bag-of-patches, and choose the one with the highest matching score as the final detection result. The position of the whole model can thus be determined by this detected bag. To achieve this, we sequentially use one of the bags to search the space. Each maintained patch in that bag is used to find its most matched patch in the searching space

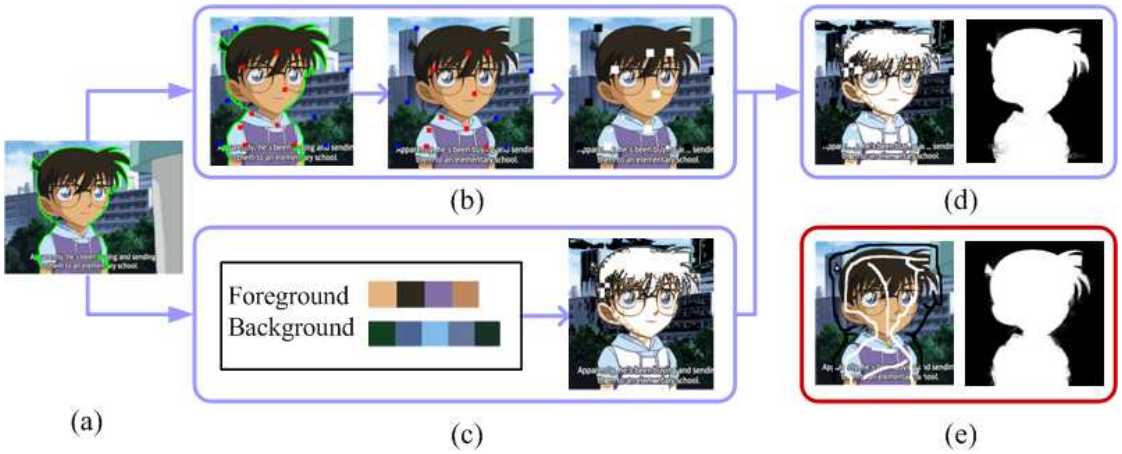


Fig. 4. Short-term coarse tracking. White regions denote foreground scribbles, while black ones denote background scribbles. (a) The boundary of the target at previous frame, (b) generating scribbles by salient point tracking, (c) generating scribbles by the discriminative color lists, (d) final scribbles and estimated alpha matte, (e) matting result by user input.

(the patch with the best SSD matching score[3]). Among those matching scores, the highest one is recorded as the matching confidence of that bag. We identify the bag with the highest confidence as the searching results. *i.e.*, the matched patch by that bag is labeled as the foreground, and the location of the model is also determined by that patch. For example, in the Book sequence of Fig. 10, BoP_7 has the highest matching confidence at Frame 720, and therefore is considered as the detected bag (the blue square). The target is then located according to BoP_7.

If the target is severely occluded, we can still infer its possible location according to previous tracking results, and the search space is the regions around that possible location. If the target reappears from complete occlusion, then searching in the whole image space may be needed. If the search space is too large, it is quite computationally expensive. Therefore, we propose a coarse-to-fine method to relocate the target. We perform search every 5 pixels and find the best one, then using gradient-based SSD matching to find the local optimum. We observed that the performance is sufficiently good in experiments by this fast method.

After locating the target, the matched patch provides a foreground scribble for matting. Discriminative color detection is further performed again to generate additional scribbles around the target. Matting thus can be successfully performed using these scribbles.

5 From Matting to Refined Tracking

In the other half loop of our system, estimated alpha mattes are first adopted to refine tracking results (*i.e.*, to obtain the accurate boundary). Each component in our model can then be sequentially updated based on the clear boundary of foreground and background.

5.1 The boundary

The alpha matte is a continuous map of foreground opacity α. The α values near the boundary of the target are hardly 0 or 1 but some values between them. Therefore, to remove such ambiguity and to obtain a clear shape boundary of the target, a certain α threshold α_T must be chosen to cut this map.

The shape boundary of the previous frame is used as the guide to determine α_T. We assume that although the target may have large deformation, its shape in two consecutive frames should not be too different.[4] Therefore, the one having the maximum likelihood with the previous shape boundary is chosen as the boundary at the current frame. We used the contour matching method [25] to calculate the likelihood because of its computational efficiency. From [25], we obtained the shape descriptor of the target in the previous frame, denoted by SD_{t-1}, and a few shape descriptors of the target boundary under different α in the current frame, denoted by SD_t^α. Then $\alpha_T = \arg\min_\alpha \|SD_t^\alpha - SD_{t-1}\|$. In practice, the alpha values are quantized into several levels. An illustrative example is given in [13].

5.2 Model updating

After obtaining the clear foreground, all the components in our model are updated.

Updating salient points. Salient points are short-term features, therefore we directly re-sample new points in the foreground and the neighboring background to obtain S_f and S_b.

3. One reason to use SSD is that it is sensitive to shift, so the estimation of the target location is accurate.

4. In addition to shape, we can also consider target size here.

Updating discriminative colors. During tracking, the background color may largely change, while the foreground may also vary due to deformation and self occlusion. Therefore, the discriminative color lists should be updated to remove invalid colors and add new discriminative colors.

Once the target is located and the boundary is estimated, we first get the color histograms of the foreground and background. Discriminative colors for the foreground and the background are then extracted respectively by calculating the log-likelihood ratio of these two histograms, as introduced in Sec. 3. For each extracted foreground discriminative color at current frame, we compare it with C_f and C_b. There are three cases:

1) C_b contains the same color, *i.e.*, one of the color features in C_b and this newly extracted discriminative color fall into the same color bin. It means that this color feature is no more discriminative for the background, and thus will be removed from C_b.
2) C_b does not contain this color while C_f does, then this color is discriminative for the foreground but already exists in C_f. No update will be performed.
3) Neither of C_b and C_f has the same color. Apparently, this color feature is a new discriminative color for the foreground and will be added to C_f.

Similarly, we compare the extracted background discriminative color with C_f and C_b. The colors in C_f which are no more discriminative are removed, and new discriminative colors for the background are added to C_b. After adaptation, the two color lists C_f and C_b have contained the newest discriminative colors as well as maintained previous color discrimination information that are still valid.

Updating the long-term model. For each bag-of-patches in the long-term model, it will be updated if and only if its corresponding position (*i.e.*, the crosspoint in the grid of the foreground, as illustrated in Fig. 3(c)), is totally visible. By that means, only the foreground is involved in model adaptation, thus avoiding model drift caused by the intervention of background regions. Once locating the model, the positions of the bags in the model are also determined. We compare the support (a $K \times K$ square, $K = 25$ in practice) of each bag with the foreground region. If the support of the bag is entirely inside the foreground, it is considered to be visible, and will be updated. Otherwise, it will be not updated at this time.

As mentioned in Sec. 3, a bag-of-patches also record the frequencies (recurrence times) of the patches in its bag. To update a bag-of-patches, we crop a new $K \times K$ patch at the bag's position, and compared it with the maintained patches by calculating their SSD. If the SSD is smaller than a predefined threshold, the cropped patch is considered very similar to that previously maintained patch. Thus the frequency of the maintained patch is increased by 1, otherwise the new patch is added to the list with initial frequency as 1. Actually it is a simple clustering process, and the frequencies of the patches are the weights of the clusters. With such a simple but efficient model adaptation approach, the long-term local appearances of the target are effectively captured and preserved.

6 Experiments

6.1 Implementation

During tracking, if the size of the target is $W \times H$, then a surrounding region with size $1.5W \times 1.5H$ is considered as its neighboring background, where salient points and discriminative color regions are extracted. We applied some morphological operators such as erosion and dilation to reduce the small noises in the scribbles.

The computational cost of our approach is mostly ascribed to the matting algorithm. We tried to provide a lot of scribbles from the discriminative colors while keeping these scribbles accurate. In a probabilistic view, we associate the i-th color with a probability value $P(i)$ indicating how likely this color is the foreground color. In the current frame, for each pixel of the i-th color, we set it as the foreground with the probability $P(i)$ if it is a distinct color for the foreground ($L_i > T_L$ in Sec. 3), or set it as the background with the probability $1 - P(i)$ if it is a distinct color for the background ($L_i < -T_L$).

The design of $P(i)$ follows a basic principle: the more discriminative the color is, the more likely this color is either foreground or background color. The log-likelihood ratio of the foreground/background color histogram L_i reflects how likely a certain color belongs to the foreground/background. For example, $L_i = 0$ means that the i-th color is the foreground color with the probability 50%, while $L_i \to +\infty$ means that the i-th color is the foreground color with the probability 100%. Therefore, we can use the cumulative distribution function (cdf) of the Gaussian distribution to characterize the probability: *i.e.*,

$$P(i) = \frac{1}{2}[1 + erf(\frac{L_i}{\sqrt{2}\sigma})], \tag{3}$$

with the parameter σ, where erf is the error function.

Figure 5 shows the scribbles that are automatically generated by our method can cover most regions at each frame, which even surpass manual labeling. In the second row of Fig. 5, white regions are those labeled as foreground, and black regions are background, while gray regions are those without labeling. Figure 6 gives us a more detailed quantitative illustration. It shows the ratio of the scribble regions to the area of the surrounding region at every frame in the

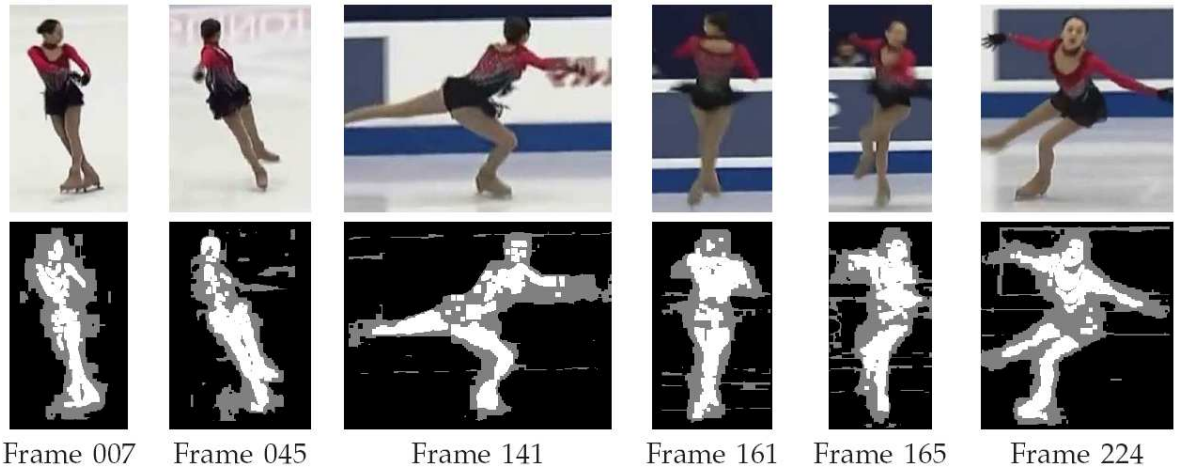
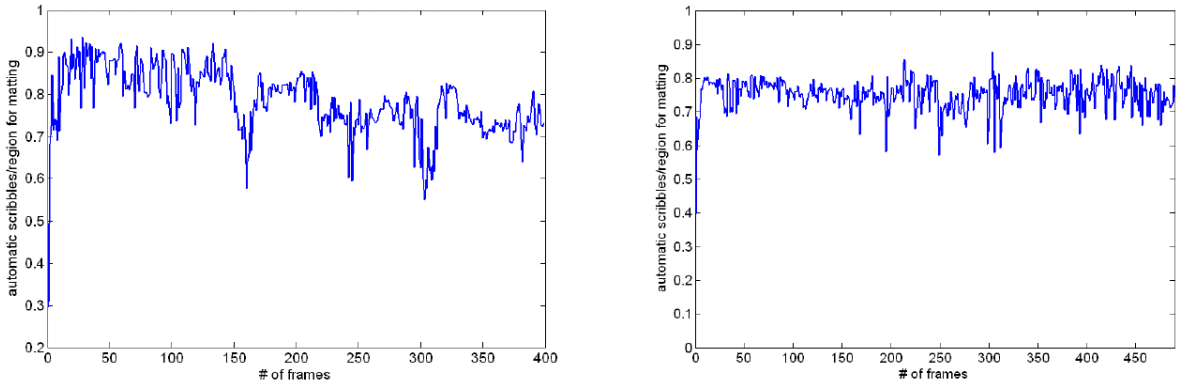


Fig. 5. The trimap shows that our method can automatically generate sufficient scribbles.



(a) Tom and Jerry (b) Bottle

Fig. 6. The area ratio of scribble regions to the surrounding regions.

Tom and Jerry sequence and Bottle sequence.

$$\text{ratio} = \frac{\#\text{ pixels of scribbles}}{\#\text{ pixels of the surrounding region}} \times 100\%. \quad (4)$$

We can see that the area ratio of scribbles keeps very high during the whole tracking process. The average area ratio of the scribbles in the Tom and Jerry sequence is 79%, while the average ratio in the Bottle sequence is 75%. Sufficient scribbles can not only guarantee good matting performance, but also significantly reduces the computational cost of matting.

The speed of our algorithm is related to the amount of the pixels with uncertain alpha values before matting, which is generally dependent on the object size. After speeding up, our method can averagely process one frame per second in Tom and Jerry sequence without code optimization in our Matlab implementation, where the object size is around 150×150. This implies a great potential of a real-time implementation in C++. As a fast matting technique [18] has been proposed recently, the computational complexity is no longer a critical issue in our algorithm.

6.2 Comparison with the baseline methods

We first compared our method with five state-of-the-art baseline tracking methods: online discriminative feature selection [10], MIL tracking [2], VTD tracking [26], Yang's method [44], and Ren's tracking-by-segmentation method [35]. Among those, Ren's method is the most closely related. Some results are shown in Fig. 7 and 8. In the Dance sequence, the dancer has large deformation throughout the entire sequence. MIL tracking has the drift problem, while our method can always keep an accurate boundary of the dancer, even if he is partially occluded (at Frame 210 and 790). In the Diving sequence, the appearances of the athlete are very different even in two consecutive frames, due to extremely fast motion. Besides, the illumination dramatically changes when he is diving to the water, as we can see from Frame 109 to 258. Accordingly, Ren's method suffered from the unreliable super-pixel matching and lost track since Frame 101. Our method can still successfully handle these complex situations and can accurately cut the body from the background.

We also provided a quantitative comparison, as we

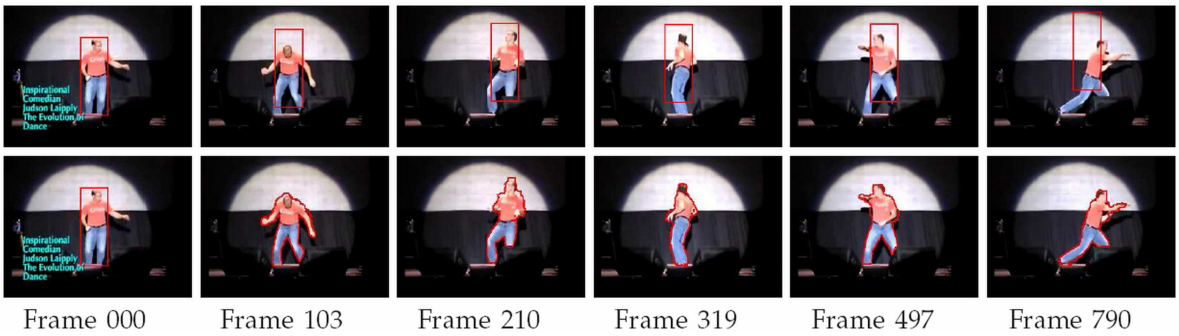


Fig. 7. Comparison with MIL tracking [2]. Top: Tracking results by MIL tracking. Bottom: Tracking results by our method.

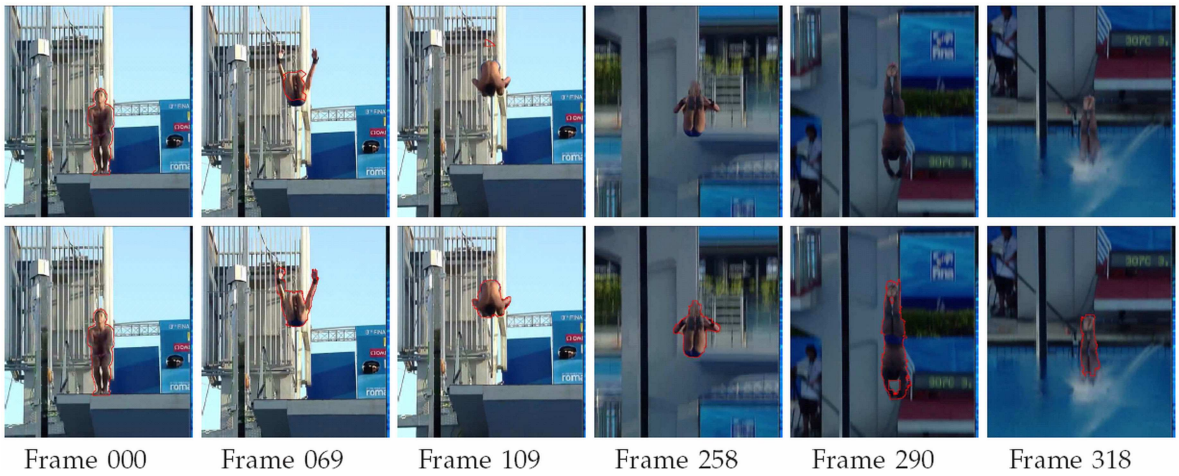


Fig. 8. Comparison with Ren's method [35]. Top: Tracking results by Ren's method. Note that in the `Diving` sequence, Ren's method lost track since Frame 101. Bottom: Tracking results by our method.

can see in Fig. 9. The ground truth of these sequences is manually labeled using a tight bounding box of the target. In our method, as we have the contour of the target, we can use a rectangular box to tightly bind this contour. The center of the bounding box is regarded as the reference position, which is then compared with the ground truth position. In Fig. 9, the tracking error of our method is smaller than the other methods. It is clear that our method shows a higher accuracy.

6.3 Comparison with video matting

Due to the page limit, we refer the readers to the conference version of this paper [13] for this comparison.

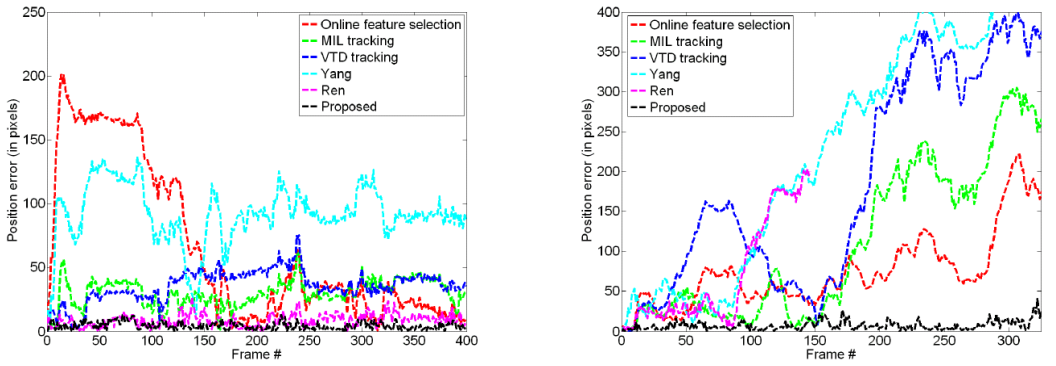
6.4 More scenarios

We further tested our method in more complex and challenging scenarios. The results are shown in Fig. 10, and more results are presented on the author's website.

The `Skating` sequence has very fast motion and significant deformation. Motion blur can be clearly observed in each frame. The background keeps changing fast, especially when the skater is jumping. Given the initialization at Frame 000, Our method performs very well and gives a clear cutout for the skater. We can even exclude the background between two legs at Frame 015.

Our method can also handle scaling and out-of-plane rotation in the `Bottle` sequence. In that sequence, the camera keeps moving and rolling. The scale of the bottle largely changes, while some part of the bottle is self-occluded. Nonetheless, the contour of the bottle is always accurately extracted in our method.

In the `Book` sequence, our method can always adaptively keep the boundary of the book, because we have a long-term model to locate the target, which can largely alleviate the errors in matting. A square on the book represents a bag-of-patches in our long-term model. Blue squares mean that these bags are not occluded and will be updated, while purple squares mean that these bags are currently under occlusion. At Frame 517, none of the bags is totally visible, the model therefore stops updating. At Frame 720, when the book reappears from complete occlusion, our long-term model can successfully relocate it.



(a) Tom and Jerry (b) Diving

Fig. 9. Quantitative Comparison with the baseline methods. Note that in the Diving sequence, Ren's method lost track since Frame 101.

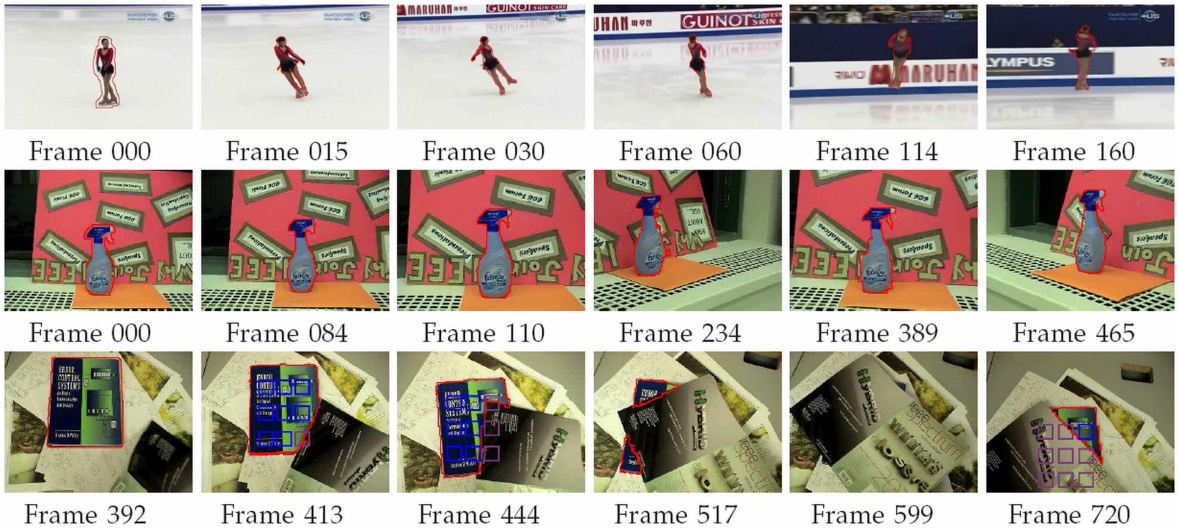


Fig. 10. More tracking results. Sequences from top to bottom: Skating, Bottle, Book.

6.5 Sensitivity

We designed two experiments to evaluate the sensitivity of the proposed method in two situations: less scribbles for matting, and false initialization of the target after full occlusion. In the first experiment, we tested the sensitivity of the scribble generation by running our algorithm with less automatic scribbles.[5] The way of generating less scribbles is that (1) we increased the threshold T_L so that some less discriminative colors are not included in the discriminative color lists, and (2) we discarded the salient points which are close to the estimated target boundary. The result is illustrated in Fig. 11.

Figure 11 shows the tracking result using two parameter configurations other than that in the previous experiment (Fig. 10). The middle row shows the situation with the least scribbles, with the corresponding scribble trimaps in the bottom row. Note that the value difference 2 between T_Ls is actually very large, because the thresholds are of the log-likelihood ratio. However, there is still a large amount of automatic scribbles in the least-scribble case, meaning that the amount of scribbles is not sensitive to these parameters. This phenomenon always hold when the target looks discriminative enough against the background. We also observe that the scribble trimaps are quite informative (Fig. 11): Even if we only look at these five trimaps (without seeing the original video), we can tell this video is about skating.

Figure 11 also reveals that our approach generally works very well using these two new parameter configurations when the background is relatively simple. There may be some inaccuracy close to boundary due to the lack of scribbles near the target boundary. When the background has similar colors as the target (Frame 116 and 127), the tracker degrades but still follows the target for the top row. In the least-scribble case, the scribbles are not enough in the trimap for Frame 116. Therefore, some red parts of the background are mistakenly treated as the foreground, and this is also the case for Frame 127. However, we observe that this

5. As the amount of automatic scribbles is always maximized in our previous experiments, we only need to test whether our approach is sensitive given less scribbles.

would not last long when the target moves away from the similar structured background, as shown in Frame 136. The target boundary is clearly split into two parts, where one part keeps tracking the target very well, and the other part can be obviously discarded by checking the visual appearance. In summary, we feel that our approach works suitably for these different parameter configurations. Although the red parts of background are misclassified in Frame 116 and 127, we argue that it may be reasonable, because it is possible this part grows from the target. In addition, even with this boundary error, the estimated target boundary is far tighter than the conventional rectangular bounding box.

In the second experiment, we examined our method when the target is erroneously initialized after full occlusion. Recall that in order to re-locate the target, we employ the long-term model to find the best match in the current frame to the one in the bags-of-patches. Once we have a good matching, this matched patch provides a foreground scribble, and the discriminative color list is activated to generate scribbles for the foregound/background. Suppose the position of the matched patch is not accurate (see the blue patch in Frame 720 of Fig. 12, we intentionally incurred the initialization errors), we then show our performance in Fig. 12.

With this false initialization, we find that the proposed approach can still estimate the target boundary very well, due to the following reason: The scribbles produced by the matched patch are on the target (see the green square in the trimap), which is correct. Furthermore, with the help of the scribbles from discriminative color lists, the tracker gathered sufficient scribble information to work correctly.

We observe that the effect brought by the false initialization is that the geometric information (the location of bags-of-patches) is not as correct as in Fig. 10. However, this geometric information would not be updated into the long-term model until all the bags-of-patches appeared in Frame 825, meaning that our long-term model is protected from such inaccurate locating. Once all the patches are not occluded (Frame 825), the geometric information is adjusted based on the current target boundary, as shown in Frame 826.

In addition to this successful example, we comment that our approach may encounter problems when the matched patch is incorrectly initialized in the background, which leads to some wrong scribbles.

6.6 Failure case

Our failure case lies in Fig. 13. Although our approach can usually obtain the accurate target boundary, and can recover from the misclassified errors robustly (Fig. 11), we observe that there are still some cases where the scribble errors may be propagated. In Fig. 13, the woman's hair contains similar color as on the book, so the positive scribbles are wrongly set on the background in Frame 81 during the fast motion of the book. The errors are then propagated in Frame 136 so that the tracker failed.

The reason causing the failure case is that the target and the background inherently contain some similar colors, which leads to (1) erroneous labels, and (2) relatively more regions with uncertain alpha values on the trimap. Fortunately, (2) can be detected by checking the ratio (Eq. 4). This indicates that we can switch to other tracking approaches when the ratio is relatively low, and hence the above failure case may be well handled.

6.7 Summary

Through the experiments, we observe that the automatic generated scribble trimap is very informative in tracking, and it indeed provides the rough location of the target. We believe that this scribble trimap can be a complementary tool for other tracking methods. For example, it can be treated as a good initialization, or help reduce the search space significantly.

7 Conclusion

This paper introduces matting into a tracking framework and proposes a closed-loop model adaptation scheme. In our framework, the scribbles for matting are automatically generated by tracking, while matting results are used to obtain accurate boundaries of the object and to update the tracking model. Our work validates the applicability of automated matting in a tracking system, and meanwhile largely avoids the model drift problem in tracking with the aid of matting results. The proposed framework can be considered as a fundamental guideline on the combination of matting and tracking. In such a framework, each component can be further explored to improve the tracking performance.

The executable code is available at the author's website:

http://www.eecs.northwestern.edu/~jfa699/code_st.html.

8 Acknowledgements

The authors thank the anonymous reviewers for their constructive comments and suggestions. This work was supported in part by National Science Foundation grant IIS-0347877, IIS-0916607, and US Army Research Laboratory and the US Army Research Office under grant ARO W911NF-08-1-0504.

References

[1] S. Avidan. Ensemble tracking. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29:261–271, 2007.

[2] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

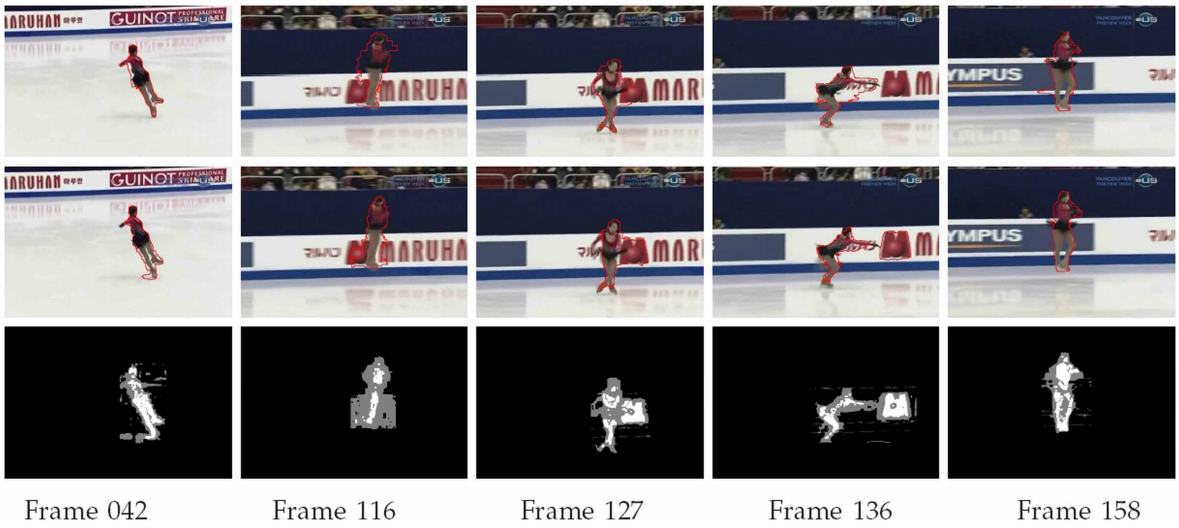


Fig. 11. Performance evaluation on `Skating` using different parameters. The parameter used in Fig. 10 is $T_L = 5$. Top: the tracking results ($T_L = 7$). Middle: the tracking results ($T_L = 9$). Bottom: the trimap shows the automatically generated scribbles ($T_L = 9$).

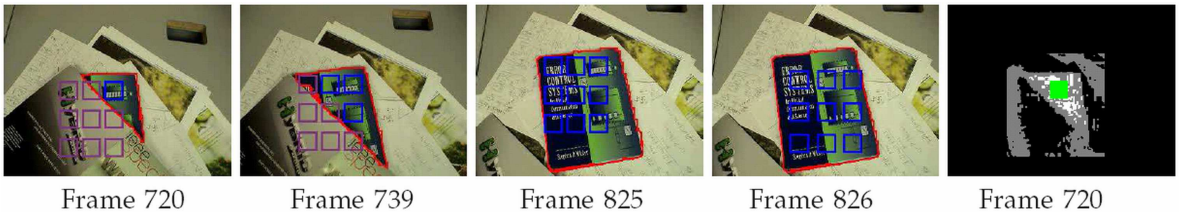
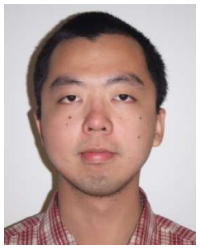


Fig. 12. The experiment of false initialization after full occlusion. The rightmost image shows the trimap for Frame 720. The green scribble is produced by the matched patch.

Fig. 13. The failure case of the proposed method. The rightmost image shows the trimap of Frame 81 (see the wrong scribbles with white color).

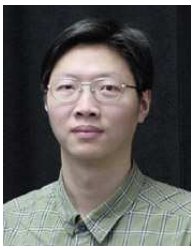
[3] X. Bai and G. Sapiro. Geodesic matting: A framework for fast interactive image and video segmentation and matting. *International Journal of Computer Vision*, 82:113–132, 2009.
[4] X. Bai, J. Wang, D. Simons, and G. Sapiro. Video snapcut: Robust video object cutout using localized classifiers. In *SIGGRAPH*, 2009.
[5] C. Bibby and I. Reid. Robust real-time visual tracking using pixel-wise posteriors. In *European Conf. Computer Vision*, 2008.
[6] C. Bibby and I. Reid. Real-time tracking of multiple occluding objects using level sets. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.
[7] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *IEEE Conf. Computer Vision and Pattern Recognition*, 1998.
[8] Y.-Y. Chuang, A. Agarwala, B. Curless, D. H. Salesin, and R. Szeliski. Video matting of complex scenes. In *SIGGRAPH*, 2002.
[9] Y.-Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski. A bayesian approach to digital matting. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2001.
[10] R. T. Collins, Y. Liu, and M. Leordeanu. On-line selection of discriminative tracking features. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27:346–352, 2005.
[11] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2000.
[12] S. Dambreville, Y. Rathi, and A. Tannenbaum. A framework for image segmentation using shape models and kernel space shape priors. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 30:1385–1399, 2008.
[13] J. Fan, X. Shen, and Y. Wu. Closed-loop adaptation for robust tracking. In *European Conf. Computer Vision*, 2010.
[14] H. Grabner and H. Bischof. On-line boosting and vision. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2006.
[15] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *European Conf. Computer Vision*, 2008.
[16] L. Grady. Random walks for image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28:1768–1783, 2006.
[17] G. Hager and P. Belhumeur. Real-time tracking of image

regions with changes in geometry and illumination. In *IEEE Conf. Computer Vision and Pattern Recognition*, 1996.
[18] K. He, J. Sun, and X. Tang. Fast matting using large kernel matting laplacian matrices. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.
[19] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.
[20] A. Jepson, D. Fleet, and T. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25:1296–1311, 2003.
[21] Z. Kalal, J. Matas, and K. Mikolajczyk. P-n learning: Bootstrapping binary classifiers from unlabeled data by structural constraints. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.
[22] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1988.
[23] M. Kim, S. Kumar, V. Pavlovic, and H. A. Rowley. Face tracking and recognition with visual constraints in real-world videos. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.
[24] P. Kohli and P. H. Torr. Measuring uncertainty in graph cut solutions. *Computer Vision and Image Understanding*, 112:30–38, 2008.
[25] F. P. Kuhl and C. R. Giardina. Elliptic fourier features of a closed contour. *Computer Graphics and Image Processing*, 18:236–258, 1982.
[26] J. Kwon and K. M. Lee. Visual tracking decomposition. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.
[27] A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. *IEEE Trans. Patten Analysis and Machine Intelligence*, 30:228–242, 2008.
[28] X. Liu and T. Yu. Gradient feature selection for online boosting. In *Int'l Conf. Computer Vision*, 2007.
[29] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.
[30] L. Lu and G. D. Hager. A nonparametric treatment on location/segmentation based visual tracking. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2007.
[31] I. Matthews, T. Ishikawa, and S. Baker. The template update problem. *IEEE Trans. Patten Analysis and Machine Intelligence*, 26:810–815, 2004.
[32] H. T. Nguyen and A. W. M. Smeulders. Robust tracking using foreground-background texture discrimination. *International Journal of Computer Vision*, 69:277–293, 2006.
[33] M. Ozuysal, V. Lepetit, F. Fleuret, and P. Fua. Feature harvesting for tracking-by-detection. In *European Conf. Computer Vision*, 2006.
[34] S. Pellegrini, A. Ess, K. Schindler, and L. V. Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *Int'l Conf. Computer Vision*, 2009.
[35] X. Ren and J. Malik. Tracking as repeated figure/ground segmentation. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2007.
[36] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *Int'l Journal of Computer Vision*, 77:125–141, 2008.
[37] T. Schoenemann and D. Cremers. Globally optimal shape-based tracking in real-time. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.
[38] S. Stalder, H. Grabner, and L. V. Gool. Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition. In *ICCV Workshop on On-line Learning for Computer Vision*, 2009.
[39] K. Toyama and A. Blake. Probabilistic tracking with exemplars in a metric space. In *Int'l Conf. Computer Vision*, 2001.
[40] J. Wang and M. F. Cohen. Image and video matting: a survey. *Foundations and Trends in Computer Graphics and Vision*, 3:97–175, 2007.
[41] J. Wang and M. F. Cohen. Optimized color sampling for robust matting. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2007.
[42] T. Woodley, B. Stenger, and R. Cipolla. Tracking using online feature selection and a local generative model. In *British Machine Vision Conference*, 2007.
[43] Y. Wu and J. Fan. Contextual flow. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2009.
[44] M. Yang, Y. Wu, and G. Hua. Context-aware visual tracking. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31:1195–1209, 2009.
[45] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 2006.
[46] Z. Yin and R. Collins. Shape constrained figure-ground segmentation and tracking. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2009.
[47] T. Yu and Y. Wu. Collaborative tracking of multiple targets. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2004.
[48] Y. Zhou and H. Tao. A background layer model for object tracking through occlusion. In *Int'l Conf. Computer Vision*, 2003.



Jialue Fan received the B.E. and M.S. degrees in electronic engineering from Tsinghua University, Beijing, China, in 2005 and 2007, respectively, and the Ph.D. degree in electrical and computer engineering from Northwestern University, Evanston, Illinois, in 2011. His research interests include computer vision and image/video processing. He was a summer intern with NEC Laboratories America (Cupertino, CA), Siemens Corporate Research (Princeton, NJ), and Adobe Systems Inc. (San Jose, CA) in 2008, 2009, and 2010, respectively. He received Walter P. Murphy Fellowship and Terminal Year Fellowship from Northwestern University in 2007 and 2010, respectively. He is a student member of the IEEE.

Xiaohui Shen is currently a Ph.D. student in the Electrical Engineering and Computer Science Department of Northwestern University. His research interests include image/video processing and computer vision. He was a summer intern with Nokia Research Center (Santa Monica, CA) and Adobe Systems Inc. (San Jose, CA) in 2010 and 2011 respectively. He received his B.S. and M.S. degrees from the Automation Department of Tsinghua University, Beijing, China, in 2005 and 2008 respectively. When he was at Tsinghua, he was awarded the First Class Outstanding Fellowship in 2003 and 2004, and the Outstanding Graduate Student Fellowship in 2006. He received Murphy Fellowship from Northwestern University in 2009. He is a student member of the IEEE.



Ying Wu received the B.S. from Huazhong University of Science and Technology, Wuhan, China, in 1994, the M.S. from Tsinghua University, Beijing, China, in 1997, and the Ph.D. in electrical and computer engineering from the University of Illinois at Urbana-Champaign (UIUC), Urbana, Illinois, in 2001.

From 1997 to 2001, he was a research assistant at the Beckman Institute for Advanced Science and Technology at UIUC. During summer 1999 and 2000, he was a research intern with Microsoft Research, Redmond, Washington. In 2001, he joined the Department of Electrical and Computer Engineering at Northwestern University, Evanston, Illinois, as an assistant professor. He is currently an associate professor of Electrical Engineering and Computer Science at Northwestern University. His current research interests include computer vision, image and video analysis, pattern recognition, machine learning, multimedia data mining, and human-computer interaction. He serves as associate editors for IEEE Transactions on Image Processing, SPIE Journal of Electronic Imaging, and IAPR Journal of Machine Vision and Applications. He received the Robert T. Chien Award at UIUC in 2001, and the NSF CAREER award in 2003. He is a senior member of the IEEE.